## Amendments to the Claims

This listing of claims replaces all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. (Original) A debugger for debugging any of a plurality of debuggees, each debuggee having a debugging type attribute selected from a plurality of debugging type attributes and representative of a type of debugging to be performed with respect to the debuggee, each debuggee also having a processor attribute selected from a plurality of processor attributes and representative of a type of processor associated with the debuggee, the debugger being instantiated on a computer and comprising:

an engine for performing debugging functions with respect to any of the plurality of debuggees, the engine including:

a plurality of debugging type blocks, each debugging type block for supporting at least one of the plurality of debugging type attributes; and

a plurality of processor blocks, each processor block for supporting at least one of the plurality of processor attributes,

wherein a particular debugging type block and a particular processor block are selected for debugging a particular debuggee based on the debugging type attribute and processor attribute of the particular debuggee.

2. (Original) The debugger of claim 1 wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type.

3. (Original) The debugger of claim 2 wherein the debugging services include services selected from a group consisting of accessing memory, accessing context, accessing system information, inserting a breakpoint, removing a breakpoint, controlling execution, and combinations thereof.

4. (Original) The debugger of claim 2 wherein the debugging type abstraction comprises programming code, and wherein at least a portion of the programming code for the debugging type abstraction is common as between at least some debugging type blocks and is shared by such debugging type blocks.

5. (Original) The debugger of claim 4 wherein the programming code for the debugging type abstraction is organized into a tree form with generic code at a base node and more specific levels of code branching out at nodes therefrom, each debugging type block including at least one node from the tree.

6. (Original) The debugger of claim 1 wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor.

7. (Original) The debugger of claim 6 wherein the processor services include services selected from a group consisting of recognizing particular processor instructions, recognizing processor states, maintaining hardware breakpoints, assembling code for the processor, disassembling code from the processor, disassembling code from a dump file produced by the processor, and combinations thereof.

8. (Original) The debugger of claim 6 wherein the processor abstraction comprises programming code, and wherein at least a portion of the programming code for the processor abstraction is common as between at least some processor blocks and is shared by such processor blocks.

9. (Original) The debugger of claim 8 wherein the programming code for the processor abstraction is organized into a tree form with generic code at a base node and more specific levels of code branching out at nodes therefrom, each processor block including at least one node from the tree.

10. (Original) The debugger of claim 1 wherein the engine further includes a high level portion for issuing generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions.

11. (Original) The debugger of claim 10 wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type, wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor, and wherein the high level portion issues generic requests to the debugging type abstraction and to the processor abstraction to accomplish debugging actions.

12. (Canceled)

13. (Canceled)

14. (Original) The debugger of claim 1 wherein the plurality of processor attributes supported by the processor blocks include processor attributes representative of members selected from a group consisting of an X86 processor family, an ALPHA processor family, an IA64 processor family, and combinations thereof.

15. (Original) The debugger of claim 1 wherein the debugger further has an executable for being executed by a user, for calling the engine, and for providing an interface between the user and the engine.

16. (Original) The debugger of claim 15 wherein the executable includes an attribute that results in the selection of a particular debugging type block in the engine.

17. (Original) The debugger of claim 15 wherein the executable includes an attribute that results in the selection of a particular processor block in the engine.

18. (Original) The debugger of claim 1 wherein the particular debuggee is a dump file produced by a processor operating a particular mode, wherein the debugging type attribute of the dump file corresponds to the particular mode, and wherein the particular debugging type block of the engine selected for debugging the dump file supports the debugging type attribute of the dump file.

19. (Original) The debugger of claim 1 wherein the particular debuggee is a dump file produced by a type of processor, wherein the processor attribute of the dump file corresponds to the type of processor, and wherein the particular processor block of the engine selected for debugging the dump file supports the processor attribute of the dump file.

20. (Original) A computer having a debugger instantiated thereon for debugging any of a plurality of debuggees, each debuggee having a debugging type attribute selected from a plurality of debugging type attributes and representative of a type of debugging to be performed with respect to the debuggee, each debuggee also having a processor attribute selected from a plurality of processor attributes and representative of a type of processor associated with the debuggee, the debugger comprising:

an engine for performing debugging functions with respect to any of the plurality of debuggees, the engine including:

a plurality of debugging type blocks, each debugging type block for supporting at least one of the plurality of debugging type attributes; and

a plurality of processor blocks, each processor block for supporting at least one of the plurality of processor attributes,

wherein a particular debugging type block and a particular processor block are selected for debugging a particular debuggee based on the debugging type attribute and processor attribute of the particular debuggee.

21. (Original) The computer of claim 20 wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type.

22. (Original) The computer of claim 21 wherein the debugging type abstraction comprises programming code, and wherein at least a portion of the programming code for the debugging type abstraction is common as between at least some debugging type blocks and is shared by such debugging type blocks.

23. (Original) The computer of claim 22 wherein the programming code for the debugging type abstraction is organized into a tree form with generic code at a base node and

more specific levels of code branching out at nodes therefrom, each debugging type block including at least one node from the tree.

24. (Original) The computer of claim 20 wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor.

25. (Original) The computer of claim 24 wherein the processor abstraction comprises programming code, and wherein at least a portion of the programming code for the processor abstraction is common as between at least some processor blocks and is shared by such processor blocks.

26. (Original) The computer of claim 25 wherein the programming code for the processor abstraction is organized into a tree form with generic code at a base node and more specific levels of code branching out at nodes therefrom, each processor block including at least one node from the tree.

27. (Original) The computer of claim 20 wherein the engine further includes a high level portion for issuing generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions.

28. (Original) The computer of claim 27 wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type, wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor, and wherein the high level portion issues generic requests to the debugging type abstraction and to the processor abstraction to accomplish debugging actions.

29. (Original) The computer of claim 20 wherein the debugger further has an executable for being executed by a user, for calling the engine, and for providing an interface between the user and the engine.

30. (Original) The computer of claim 29 wherein the executable includes an attribute that results in the selection of a particular debugging type block in the engine.

31. (Original) The computer of claim 29 wherein the executable includes an attribute that results in the selection of a particular processor block in the engine.

32. (Original) A method comprising:

determining, for a particular debuggee, a debugging type attribute of the particular debuggee;

selecting a particular debugging type block of an engine of a debugger for debugging the particular debuggee based on the determined debugging type attribute;

determining, for the particular debuggee, a processor attribute of the particular debuggee;

selecting a particular processor block of the engine of the debugger for debugging the particular debuggee based on the determined processor attribute; and

employing the selected debugging type block and the selected processor block to debug the particular debuggee.

33. (Original) The method of claim 32 wherein determining the debugging type attribute comprises receiving a selection of a particular type of debugging from a user.

34. (Original) The method of claim 32 wherein determining the processor attribute comprises sensing a particular type of processor from the debuggee.

35. (Original) The method of claim 32 wherein determining the debugging type attribute comprises sensing a particular type of debugging from the debuggee.

36. (Original) The method of claim 32 further comprising employing a high level portion of the engine of the debugger to issue generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions.

37. (Original) The method of claim 36 wherein employing the high level portion comprises issuing generic requests from the high level portion to a debugging type abstraction and to a processor abstraction to accomplish debugging actions, the debugging type abstraction

comprising a plurality of debugging type blocks and being available to provide debugging type services that vary in implementation for each debugging type, the processor abstraction comprising a plurality of processor blocks and being available to provide processor services that vary in implementation for each processor.

38. (Original) The method of claim 32 further comprising running an executable of the debugger in response to a command from a user, the executable for calling the engine and for providing an interface between the user and the engine.

39. (Original) The method of claim 38 wherein the executable includes an identification of the debugging type attribute of the debuggee, the method comprising selecting the particular debugging type block in the engine based on the identification.

40. (Original) The method of claim 38 wherein the executable includes an identification of the processor attribute of the debuggee, the method comprising selecting the particular processor block in the engine based on the identification.

41. (Canceled)

42. (Original) A computer-readable medium having computer-executable instructions thereon, the instructions being organized into modules comprising:

a first module for determining, for a particular debuggee, a debugging type attribute of the particular debuggee;

a second module for selecting a particular debugging type block of an engine of a debugger for debugging the particular debuggee based on the determined debugging type attribute;

a third module for determining, for the particular debuggee, a processor attribute of the particular debuggee;

a fourth module for selecting a particular processor block of the engine of the debugger for debugging the particular debuggee based on the determined processor attribute; and

a fifth module for employing the selected debugging type block and the selected processor block to debug the particular debuggee.

43. (Original) The medium of claim 42 wherein the first module determines the debugging type attribute by receiving a selection of a particular type of debugging from a user.

44. (Original) The medium of claim 42 wherein the third module determines the processor attribute by sensing a particular type of processor from the debuggee.

45. (Original) The medium of claim 42 wherein the first module determines the debugging type attribute by sensing a particular type of debugging from the debuggee.

46. (Original) The medium of claim 42 further comprising a sixth module for employing a high level portion of the engine of the debugger to issue generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions.

47. (Original) The medium of claim 46 wherein the sixth module issues generic requests from the high level portion to a debugging type abstraction and to a processor abstraction to accomplish debugging actions, the debugging type abstraction comprising a plurality of debugging type blocks and being available to provide debugging type services that vary in implementation for each debugging type, the processor abstraction comprising a plurality of processor blocks and being available to provide processor services that vary in implementation for each processor.

48. (Original) The medium of claim 42 further comprising a sixth module for running an executable of the debugger in response to a command from a user, the executable for calling the engine and for providing an interface between the user and the engine.

49. (Original) The medium of claim 48 wherein the executable includes an identification of the debugging type attribute of the debuggee, the second module selecting the particular debugging type block in the engine based on the identification.

50. (Original) The medium of claim 48 wherein the executable includes an identification of the processor attribute of the debuggee, the fourth module selecting the particular processor block in the engine based on the identification.